

JFFS2へのXATTR実装とSELinux

CE Linux Forum Technical Jamboree (Oct 27, 2006)

NEC OSS推進センタ

海外 浩平

<kaigai@ak.jp.nec.com>

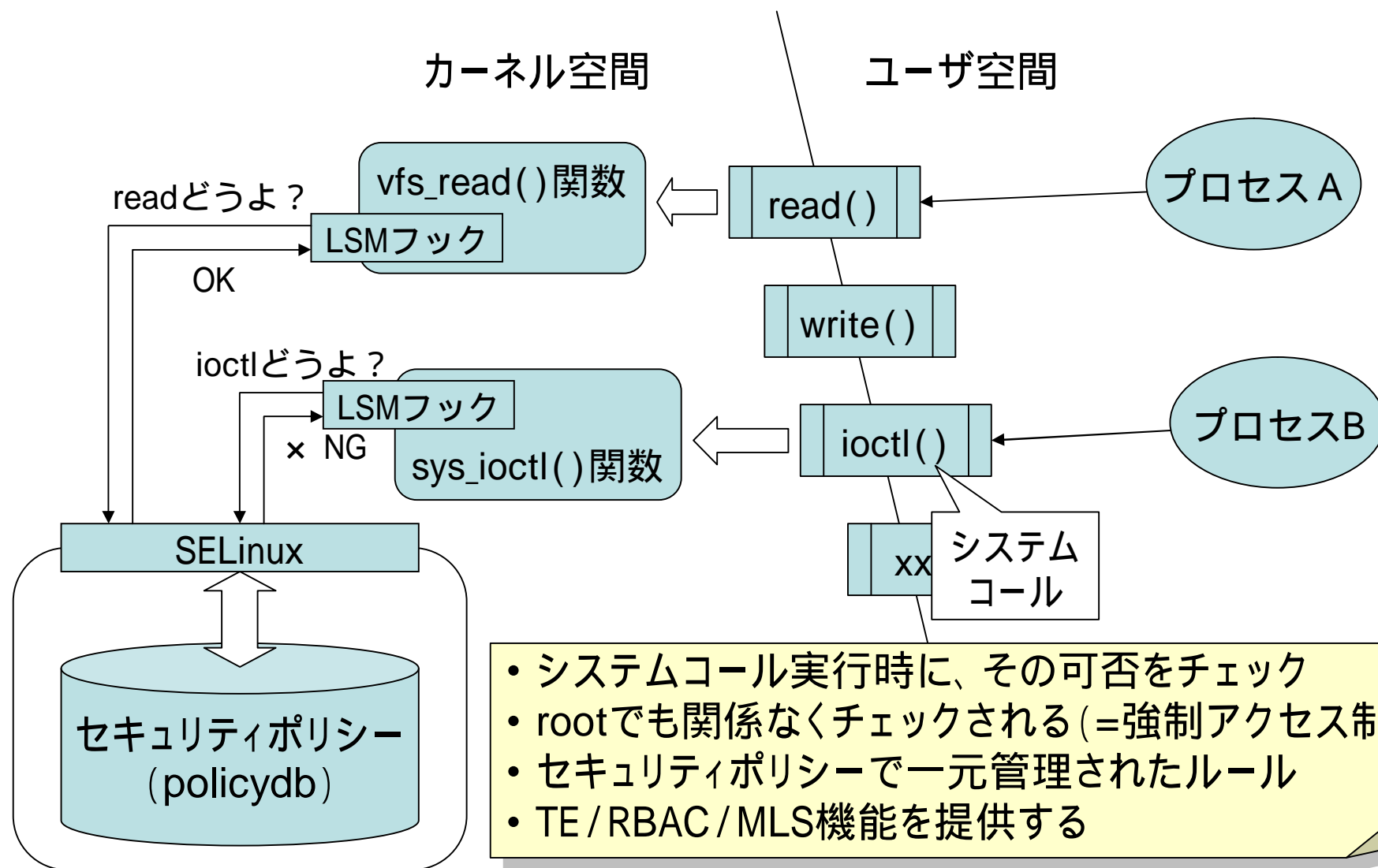
目 次

- インTRODクシヨN
- XATTR / JFFS2の構造
- コミユニティからのフィードバック
- XATTR / JFFS2の活用と今後

そもそも何をやっている人ですか？

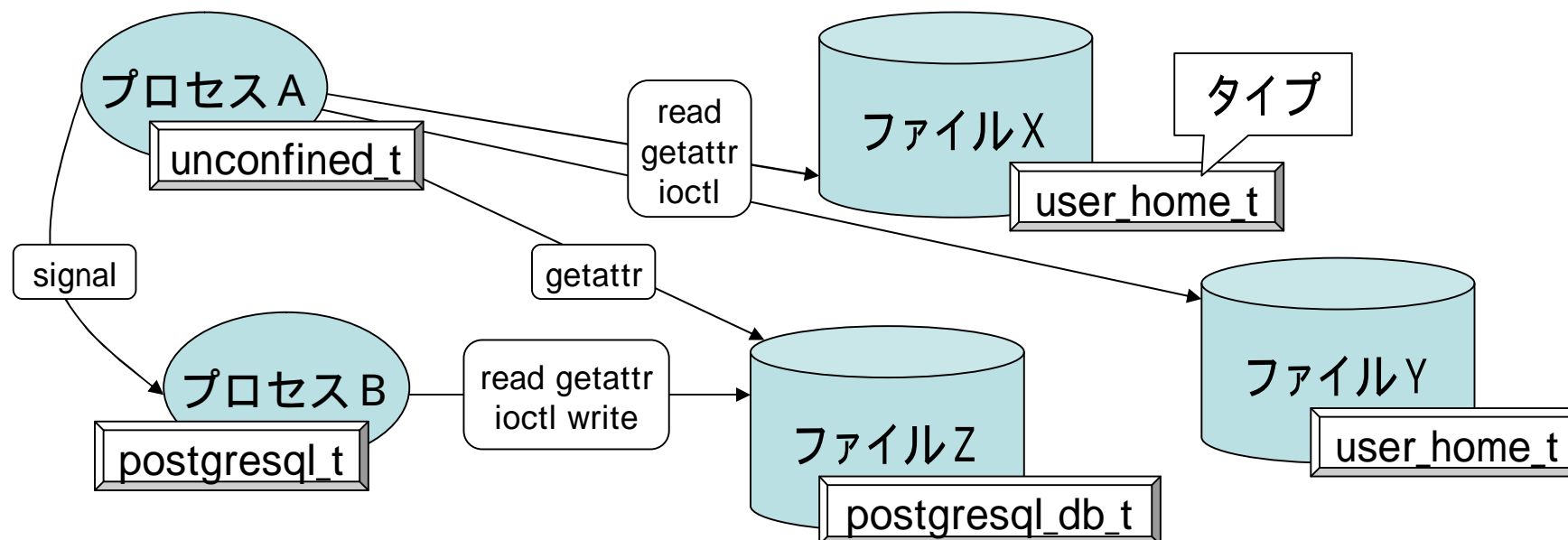
- SELinux周りの開発をしています。
 - SELinuxのスケラビリティ改善(2004)
 - ✓ 昔のSELinuxはパフォーマンスが出なかった
 - ✓ RCUを使ってロックレス参照を可能にした
 - JFFS2のXATTR対応(2005-2006)
 - ✓ JFFS2 = Flash-ROM用のファイルシステム
 - ✓ SELinuxを使うにはXATTRの対応が必須
 - SELinux対応PostgreSQLの開発(進行中)
 - ✓ OSと一体化した情報フロー制御

その前に、SELinuxって何ですか？(1)



- システムコール実行時に、その可否をチェック
- rootでも関係なくチェックされる (=強制アクセス制御)
- セキュリティポリシーで一元管理されたルール
- TE/RBAC/MLS機能を提供する

その前に、SELinuxって何ですか？(2)



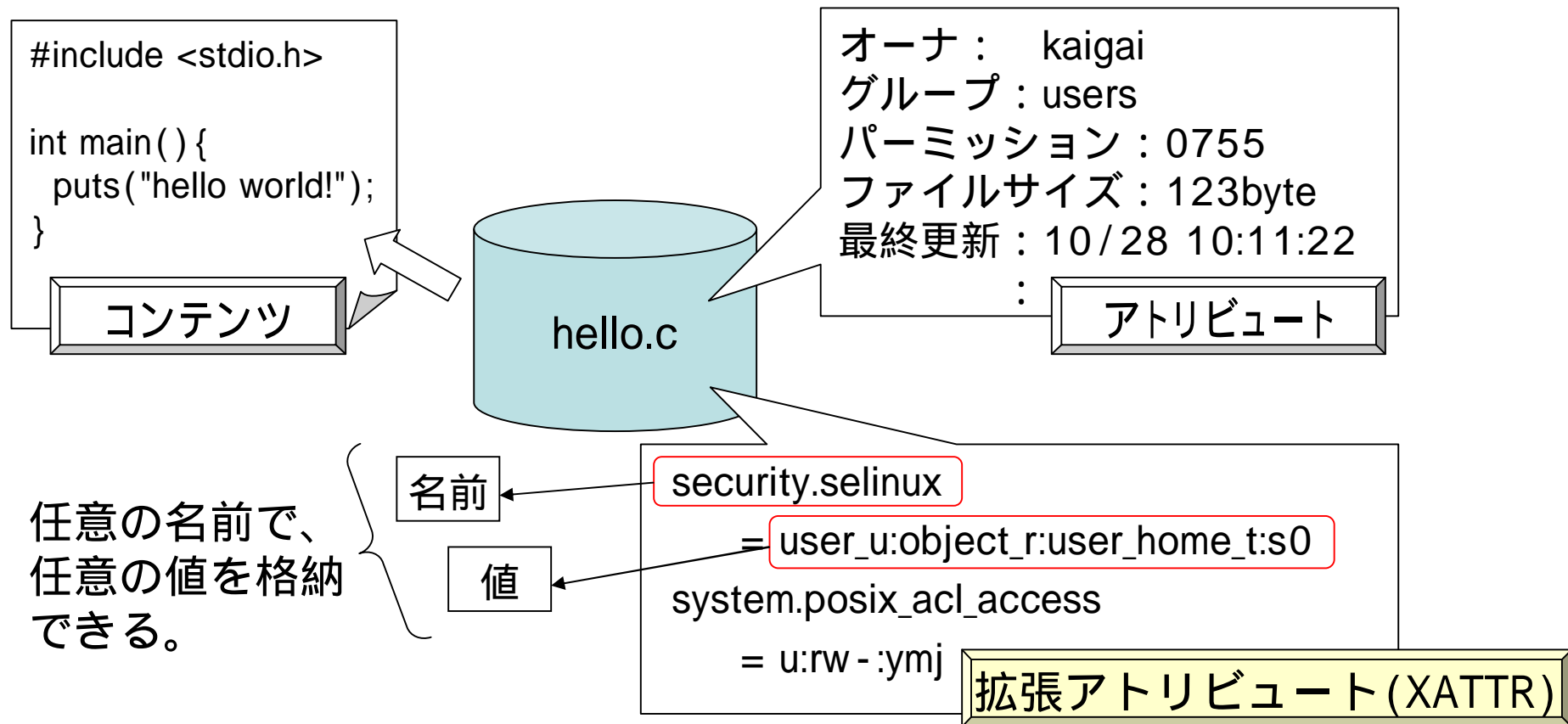
セキュリティポリシー

```
allow unconfined_t user_home_t:file {read getattr ioctl};
allow unconfined_t postgresql_db_t:file {getattr};
allow postgresql_t postgresql_db_t:
    file {read getattr ioctl write};
allow unconfined_t postgresql_t:process {signal};
```

全てのプロセス・リソースに
タイプを付与する。
"型"の強制、即ち
TE (Type Enforcement)

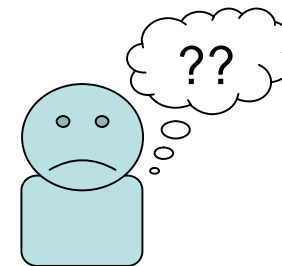
その前に、XATTRって何ですか？

- XATTR=eXtended ATTRibute (拡張アトリビュート)



JFFS2のXATTR対応 ～ 開発のスタート(1)～

- 組み込み分野とSELinuxの親和性
 - セキュリティパッチのリリースがあっても
 - 即座にパッチ適用ができないケースもある
 - 同一構成のマシンが大量に出回る
 - 0-day攻撃の脅威が予測不能
 - 生命・財産に対する脅威に直結する
- では、何が必要になるのか？
 - busyboxのSELinux対応コマンド
 - セキュリティポリシーのサイズ削減
 - ディスクレスファイルシステムでのXATTR対応



JFFS2のXATTR対応 ～ 開発のスタート(2)～

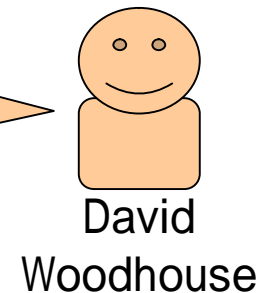
- 本家コミュニティの開発動向は？
 - 実際に聞いてみた。

Jul, 2005@Ottawa Linux Symposiumにて

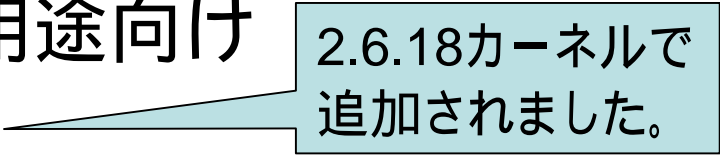


1. busyboxはRussell Cockerのパッチがあったはず
2. ポリシーのサイズ削減は、丁度開発してるところ
3. JFFS2のXATTRは、隠しファイルを作って実装した人がいたみたいだけども...

JFFS2のXATTR実装は今のところ計画していないけれども、コントリビューションは大歓迎。



XATTRの対応状況 (2.6.18カーネル)

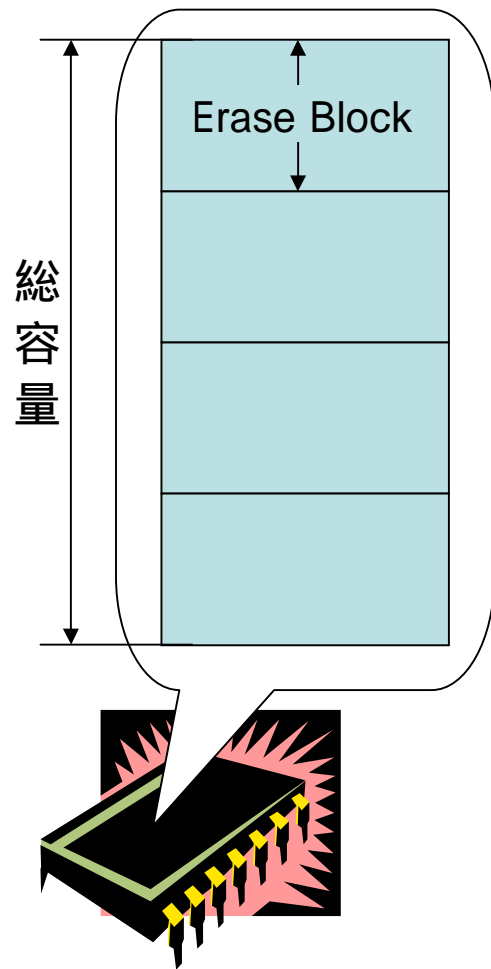
- サーバ/WS向け
 - Ext2, Ext3, XFS, ReiserFS, JFS
- 組み込み用途向け
 - JFFS2 

2.6.18カーネルで追加されました。
 - Flash-ROM等を有効利用するために開発されたFS
 - ロバストネスとメディアの寿命を重視した設計
- ネットワークファイルシステム
 - NFSv3, NFSv4, CIFS (但し、POSIX-ACLのみの対応)
- 擬似ファイルシステム
 - FUSE

目 次

- イン트로ダクション
- XATTR / JFFS2の構造
- コミュニティからのフィードバック
- XATTR / JFFS2の活用と今後

JFFS2ってどんなファイルシステム？(1)



- Flash - ROMの特性

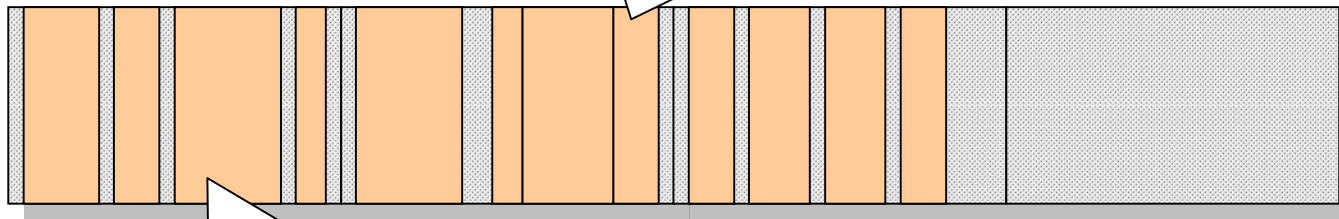
- ✓ MTDデバイスとして認識される
- ✓ 書き込み回数に制限。およそ10万回程度
- ✓ 一度書き込まれたデータは、Erase Block単位でしか消去できない。

- JFFS2の特徴

- ✓ MTDデバイスの上位レイヤとして実装
- ✓ メディア上の特定箇所を酷使しない工夫
(追記型アーキテクチャ)
- ✓ ガベージコレクタがメディア上のデータを再配置し、Erase Block単位の消去を可能に。

JFFS2ってどんなファイルシステム？(2)

追記型アーキテクチャ

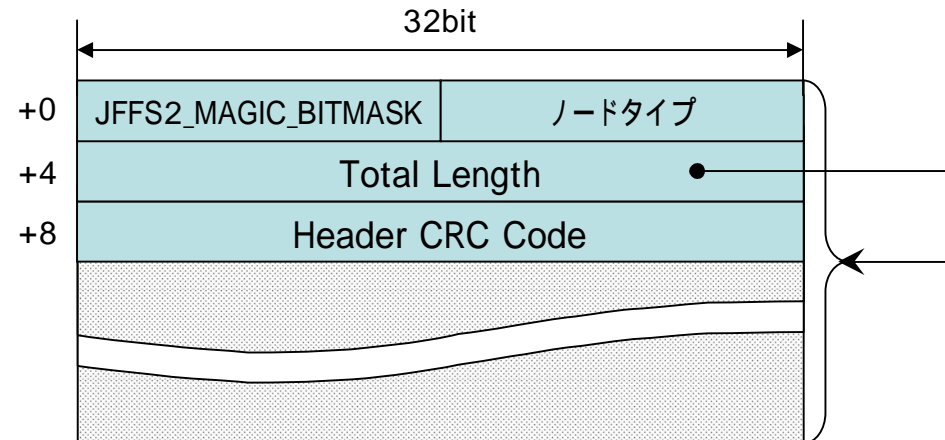


書き込み単位：ノード

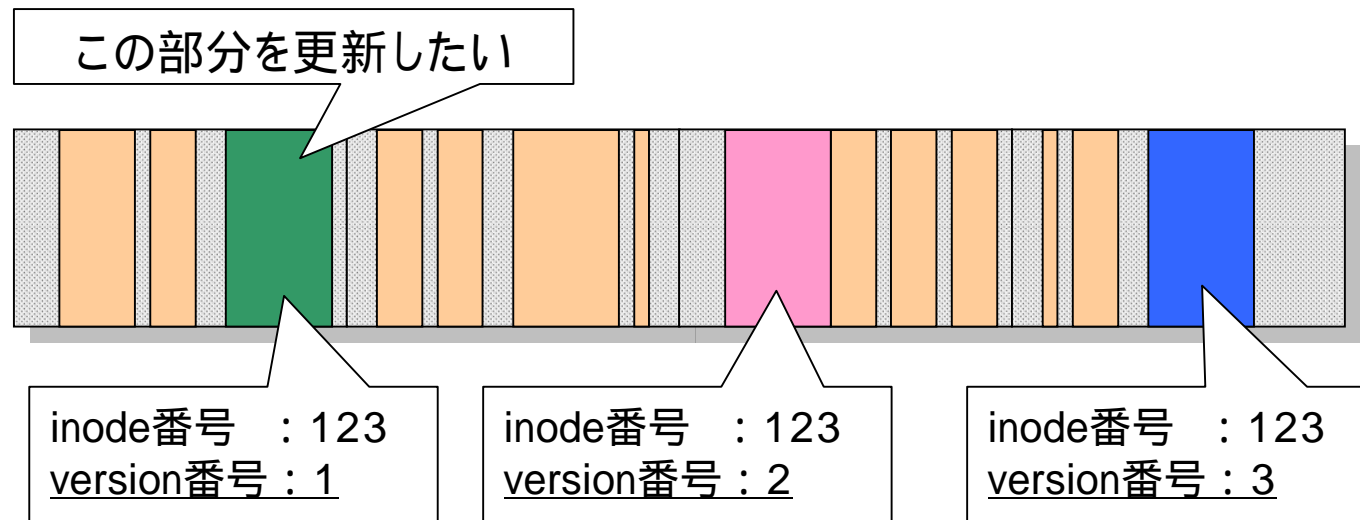
- 任意の長さで記録できる
- CRCコードで整合性を確認
- inode型など7種類

JFFS2_NODETYPE_INODE
JFFS2_NODETYPE_DIRENT
JFFS2_NODETYPE_CLEANMARKER
JFFS2_NODETYPE_PADDING
JFFS2_NODETYPE_SUMMARY
JFFS2_NODETYPE_XATTR
JFFS2_NODETYPE_XREF

ノード共通ヘッダ

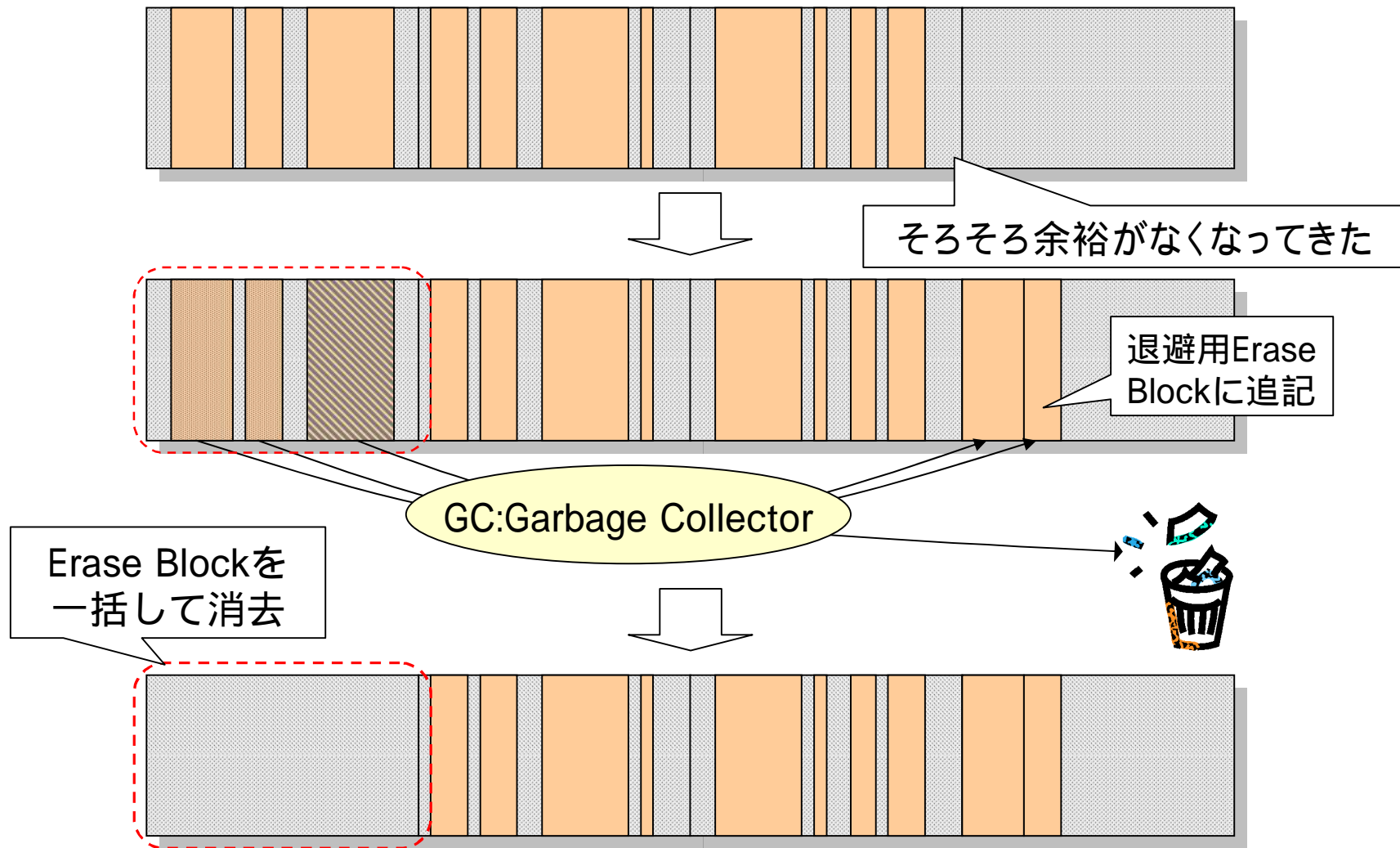


JFFS2ってどんなファイルシステム？(3)

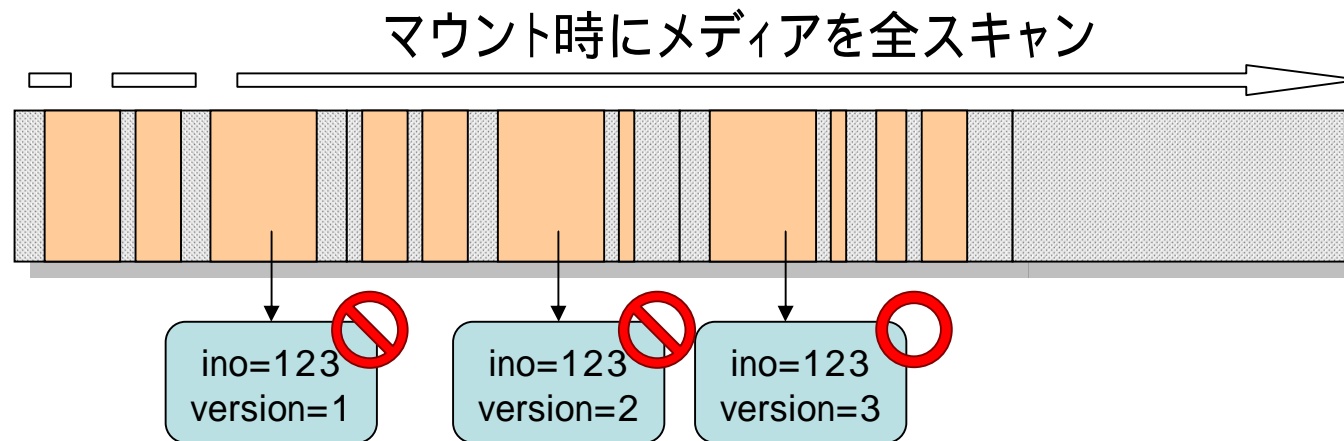


- 更新 = version番号のインクリメント + 追記
- inode番号が同じで、version番号の異なる複数のノードが存在
- 古いversion番号のノードは無視される
やがてGCによって回収され、Erase Blockの消去後に領域は
利用可能となる。

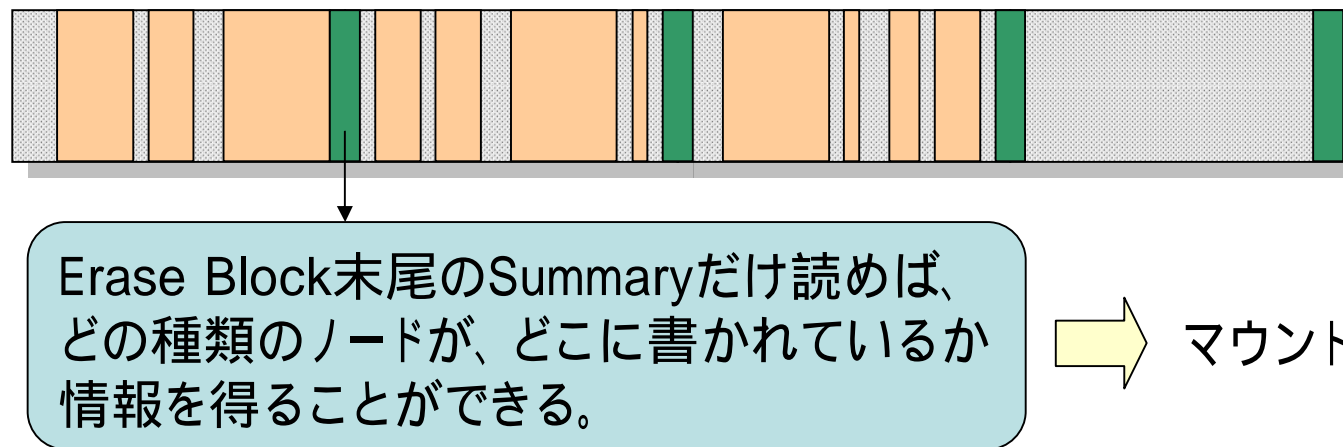
JFFS2ってどんなファイルシステム？(4)



JFFS2ってどんなファイルシステム？(5)



EBS(Erase Block Summary)機能

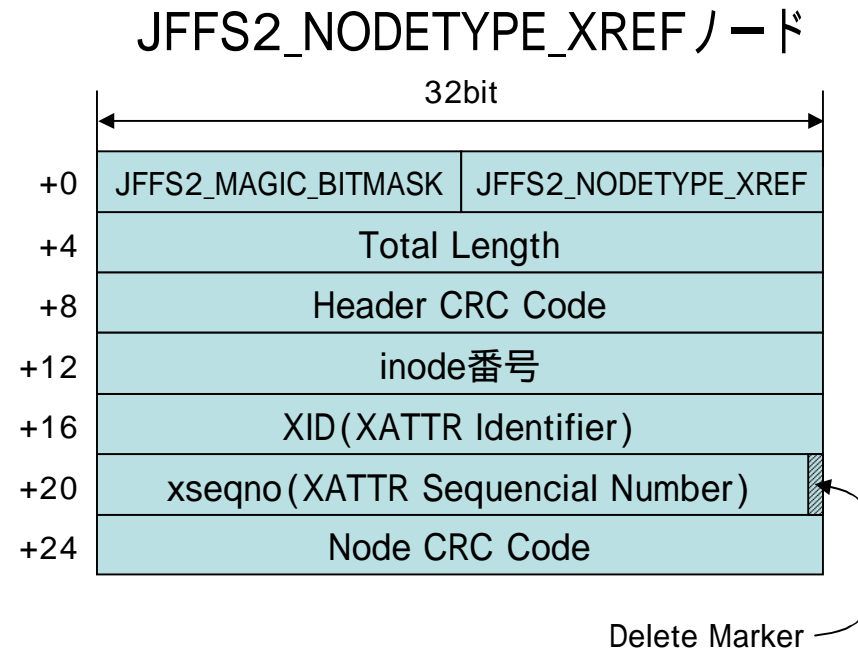
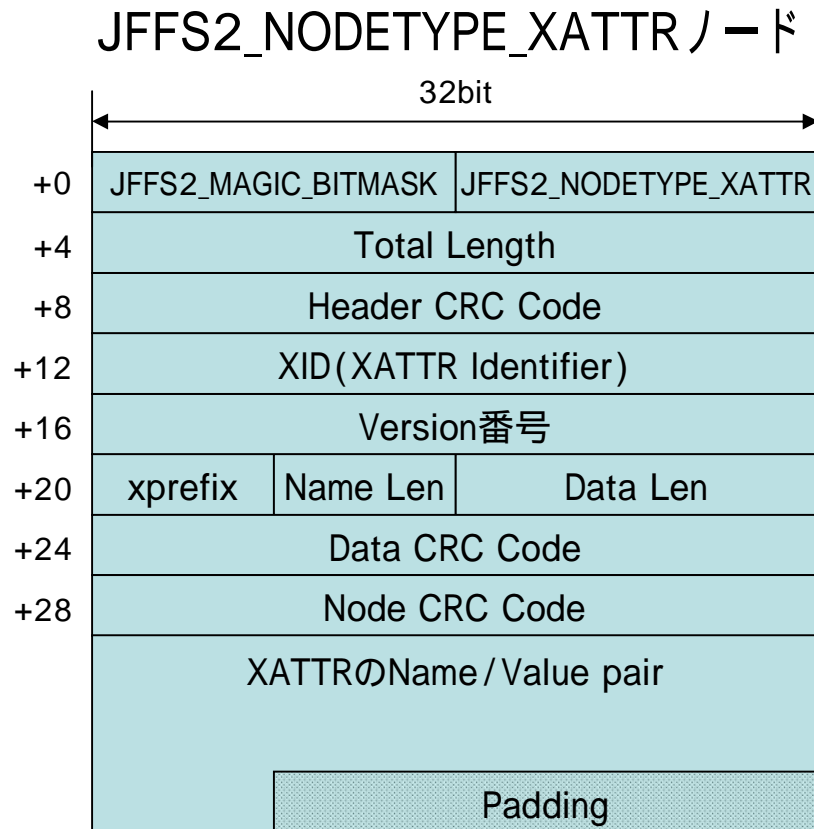


JFFS2ってどんなファイルシステム？(6)

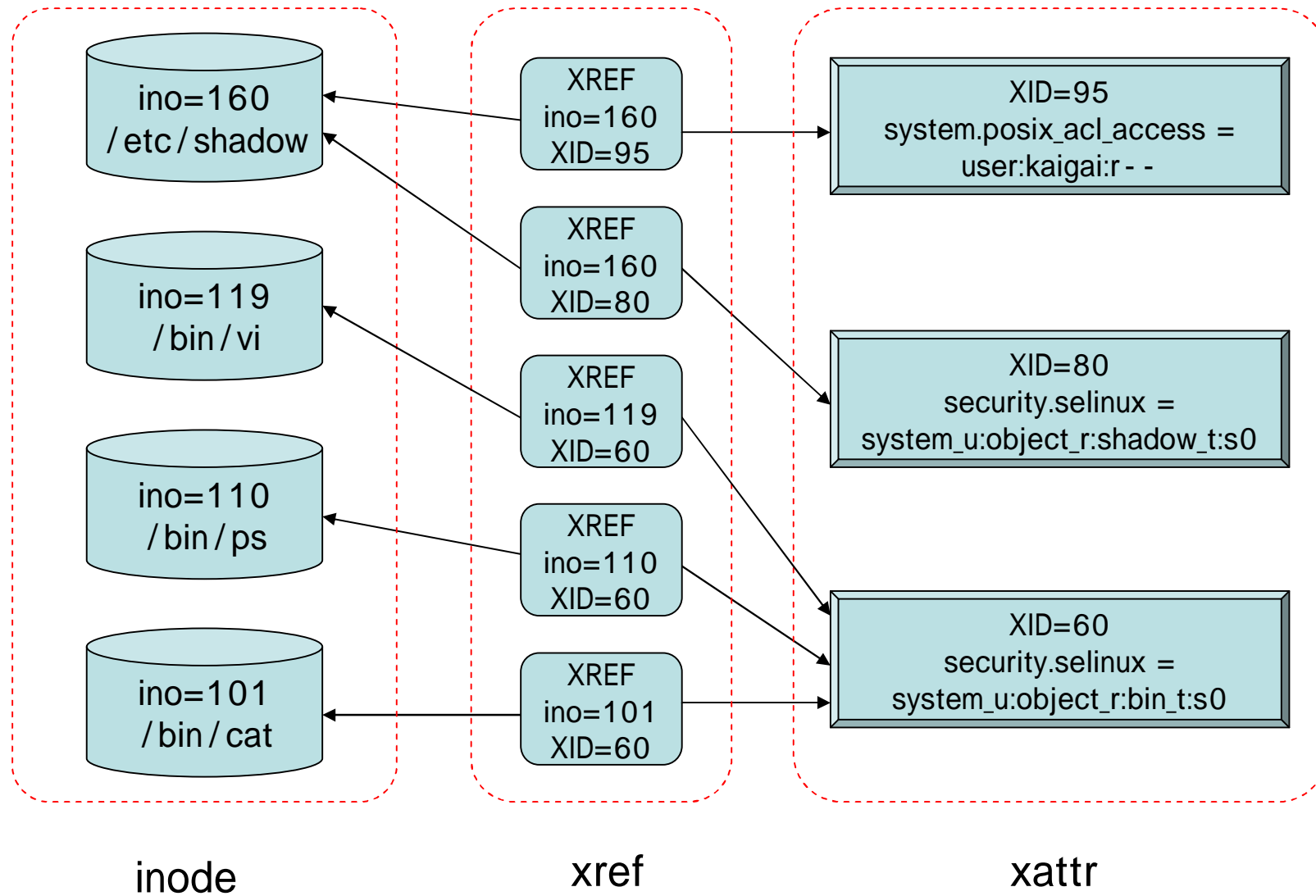
- 追記型アーキテクチャ
 - メディア上の特定箇所を酷使しない。 長保ち！
 - Flash-ROMの特性に合わせ、GCがノードを再配置
- ロバスト性
 - 特定位置にSuper Blockを持たない
 - データの読み込み時にCRCチェックを実行する
- 欠点も、、、
 - FSのマウント時には、メディアの全スキャンが必要
 - EBS(Eraseblock Summary Support)でマシになった

XATTR on JFFS2 の基本構造(1)

- XATTRを格納するノード型の定義



XATTR on JFFS2 の基本構造 (2)

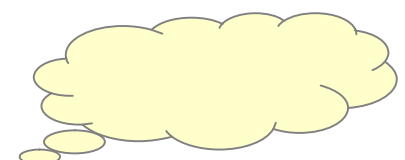


XATTR on JFFS2 の基本構造 (3)

- inodeとxattrがN:M関係を持つ
 - xattrのname / valueペアを複数のinodeが共有する
 - ストレージ / メモリ利用効率で優れる
- 既存のノード構造には手を加えない
 - 古いカーネルでは、単に「未知のノード」として無視される
- 格納することができるのは
 - SELinuxラベル (security.*)
 - POSIX ACL (system.posix_acl_{access default})
 - 任意のユーザ型データ (user.* , trusted.*)

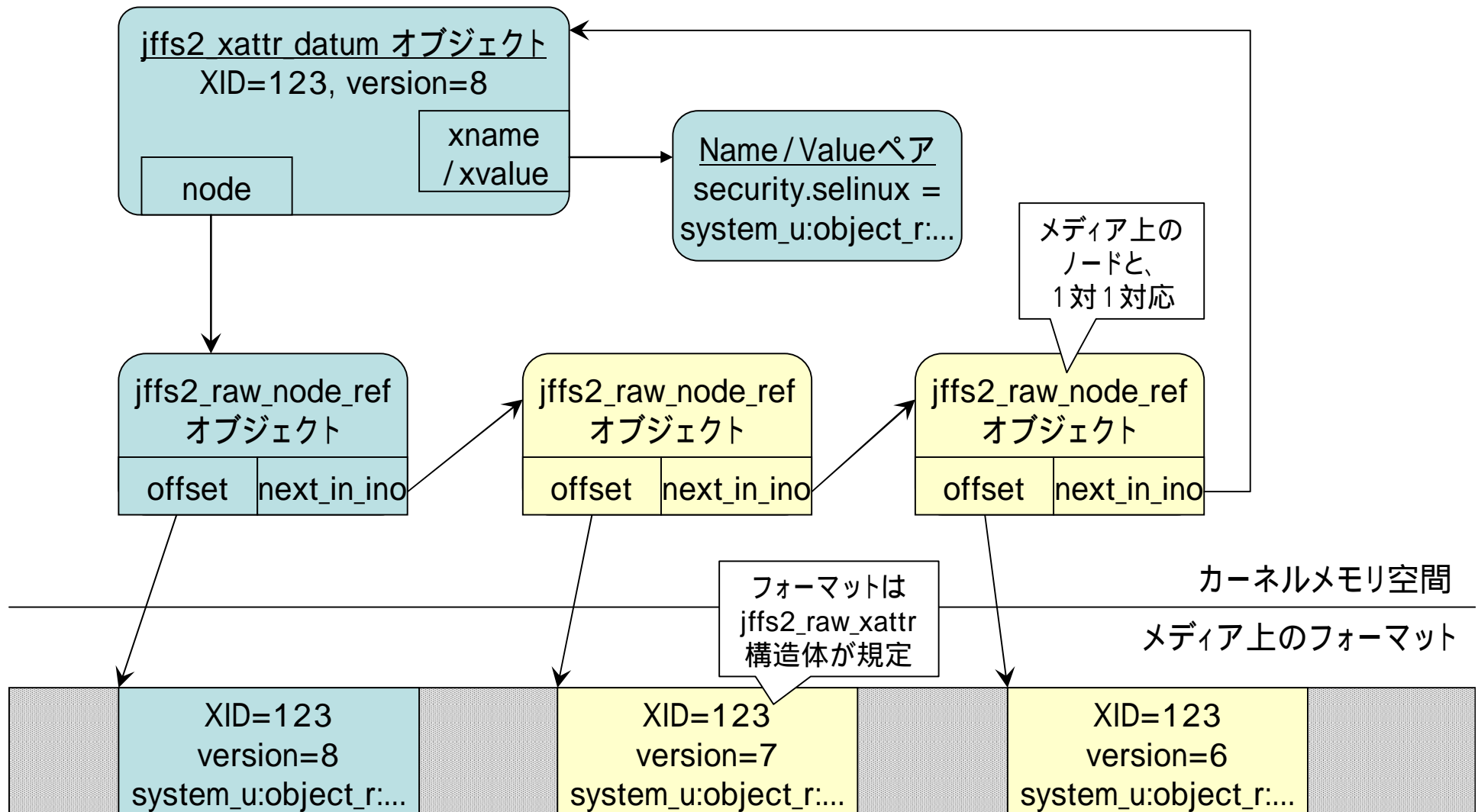
過去に提案された方法

- 隠しファイルを利用した方法
 - Lorenzo Hernández García - Hierro氏
 - ReiserFSで利用されている方法と同じ
 - JFFS3のロードマップに載っているから & バグが取れなくて困っているという理由で投げ出してしまった....。
- ディレクトリエントリの拡張
 - Ma Yun氏
 - 私がパッチを投稿する6時間前(！)にポストされた。
 - ディレクトリエントリの中にXATTRのName / Valueペアを入れる。
 - 同一のName / Valueペアを共有できない。
 - SELinuxでは、非常に多くの同一Name / Valueペアが使われる。
 - JFFS2_NODETYPE_DIRENTノードの意味を変えるのは望ましくない。

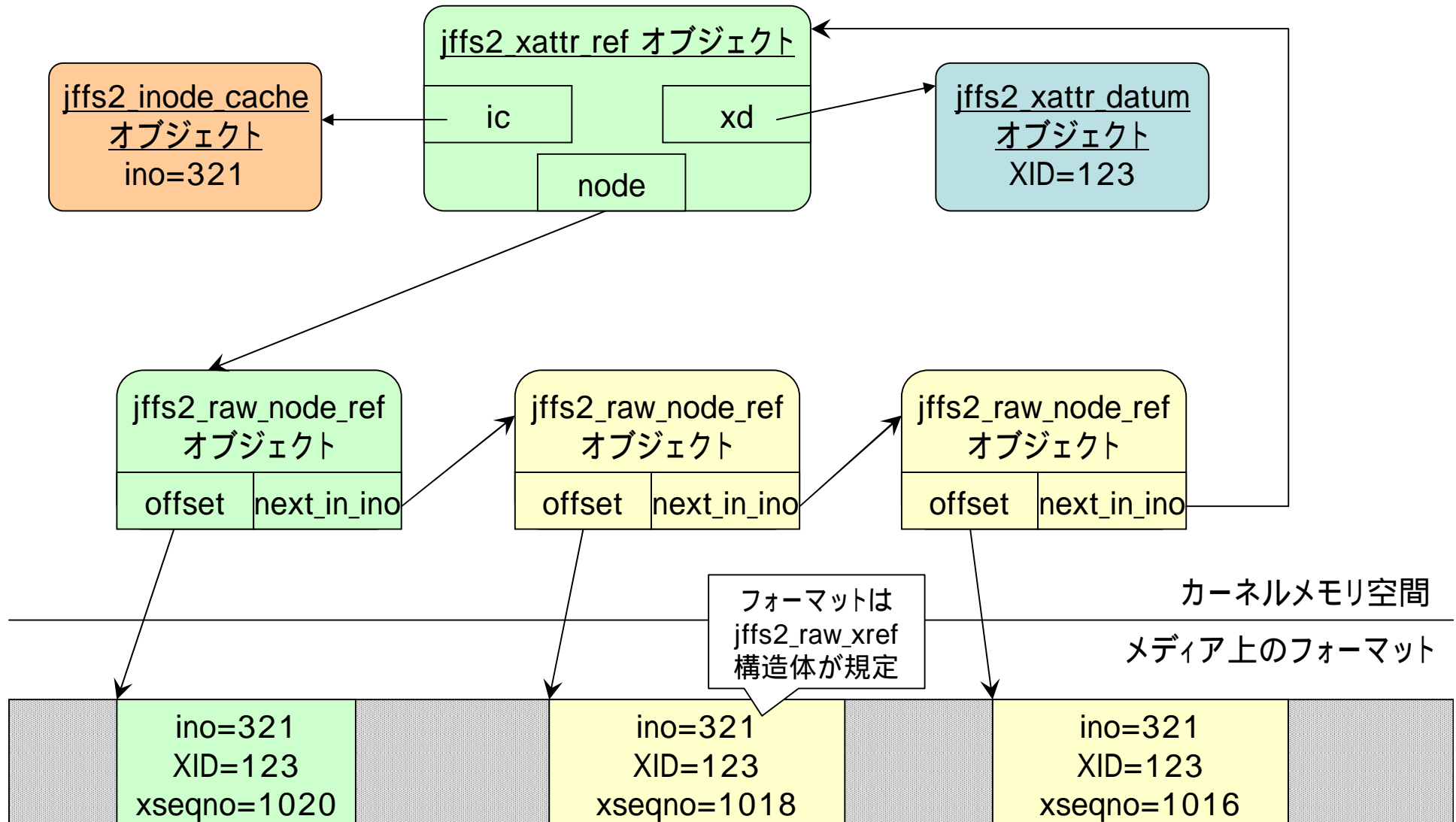


特許じゃなくてよかった。

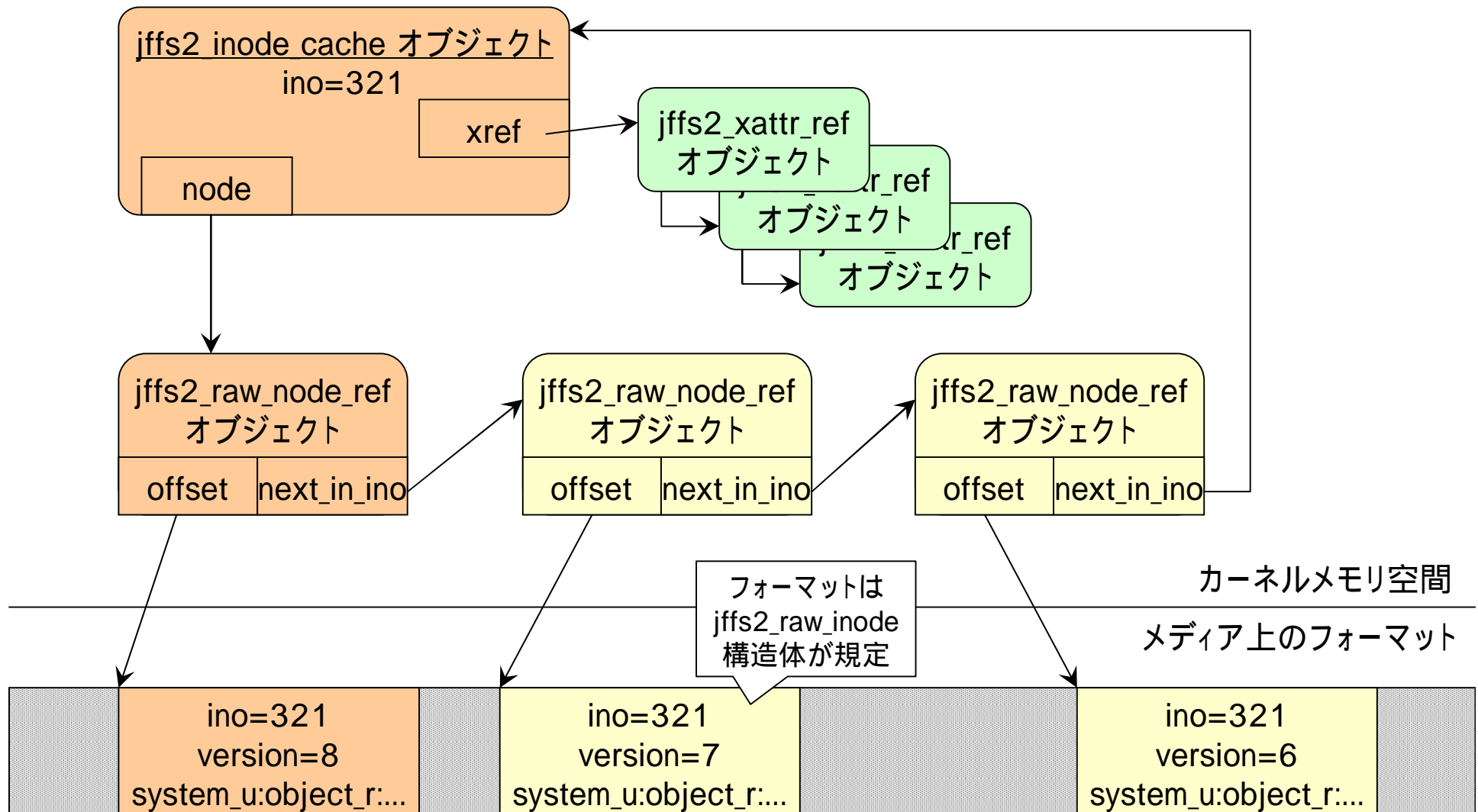
XATTR on JFFS2 のデータ構造(1)



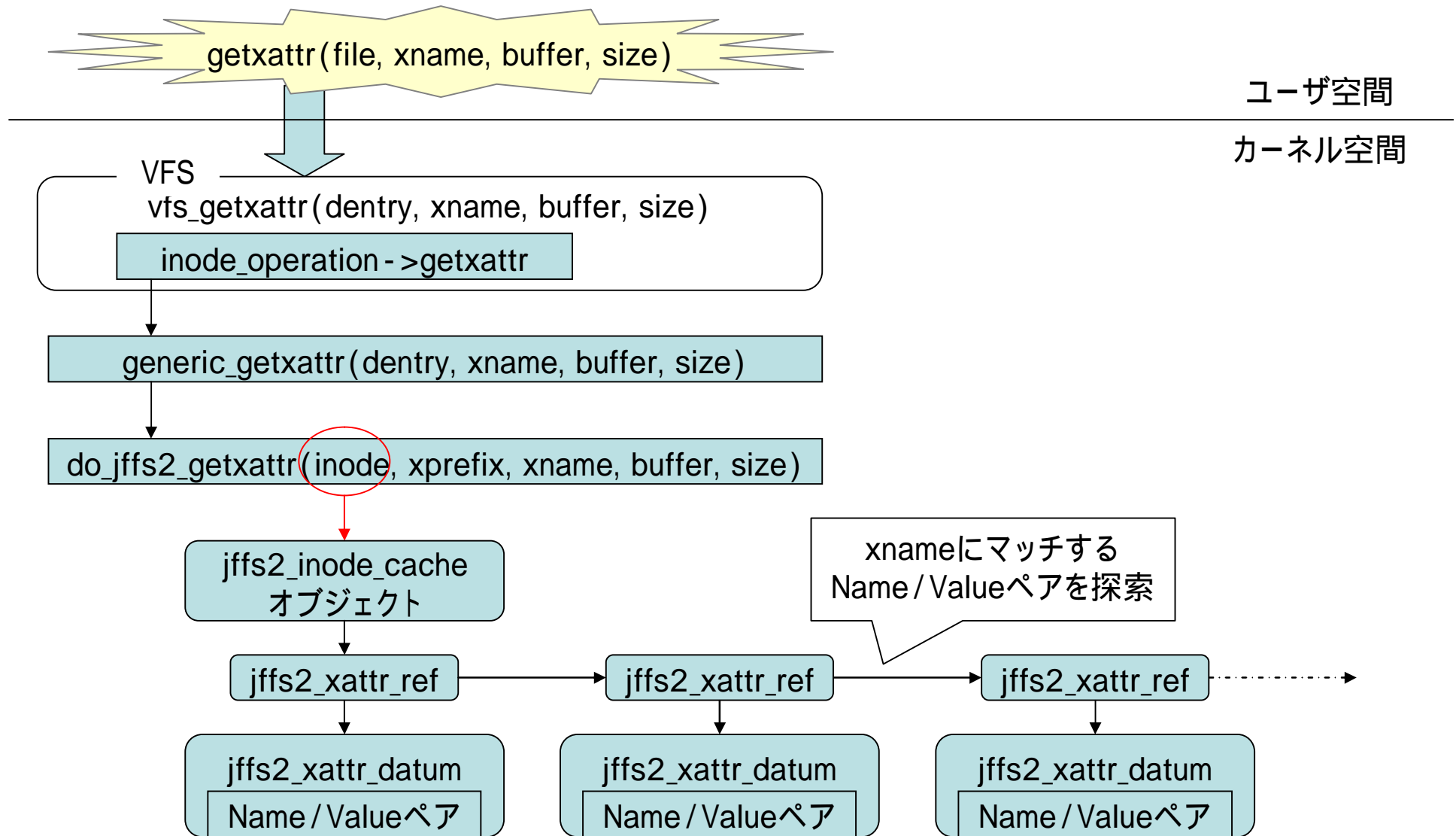
XATTR on JFFS2 のデータ構造(2)



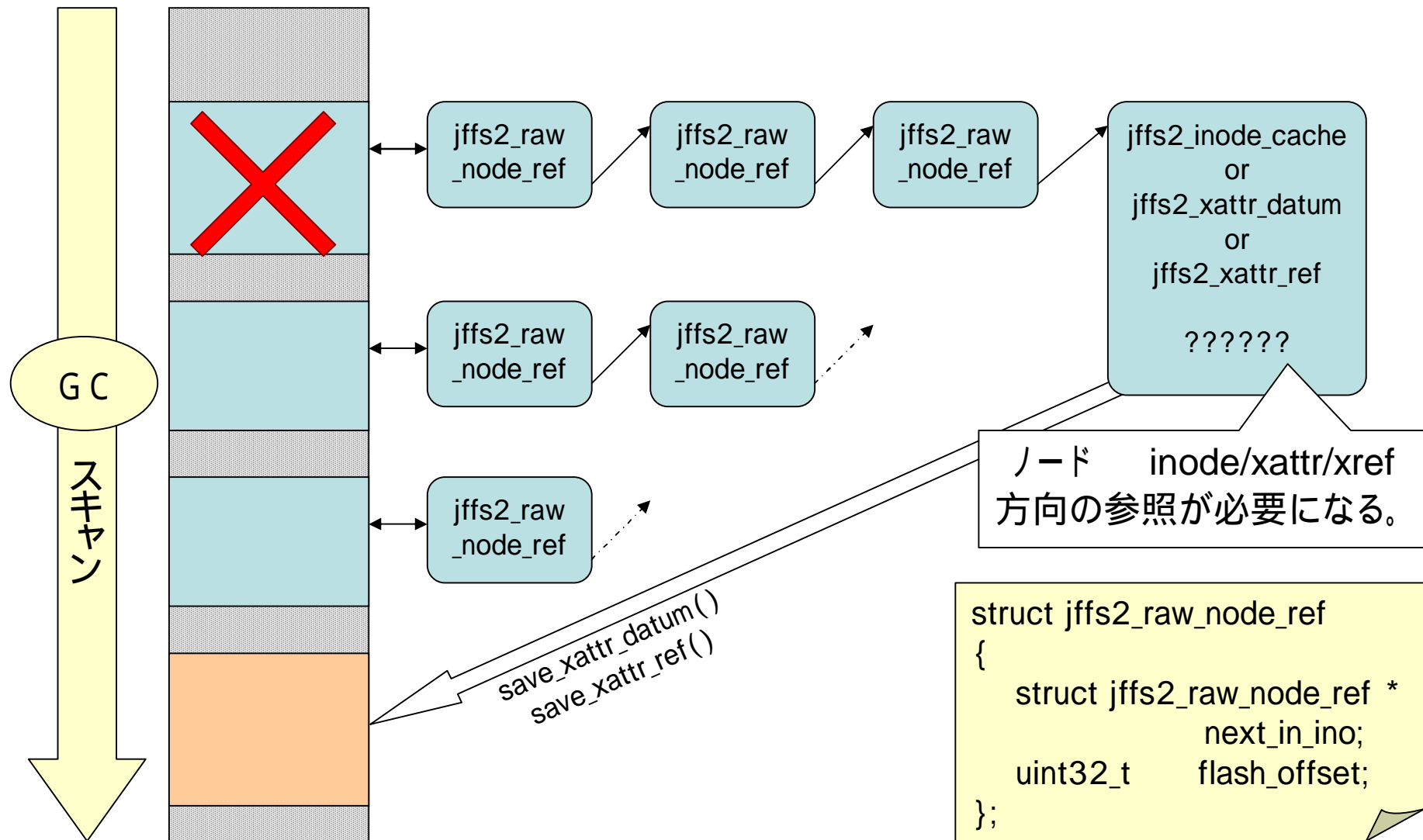
XATTR on JFFS2 のデータ構造 (3)



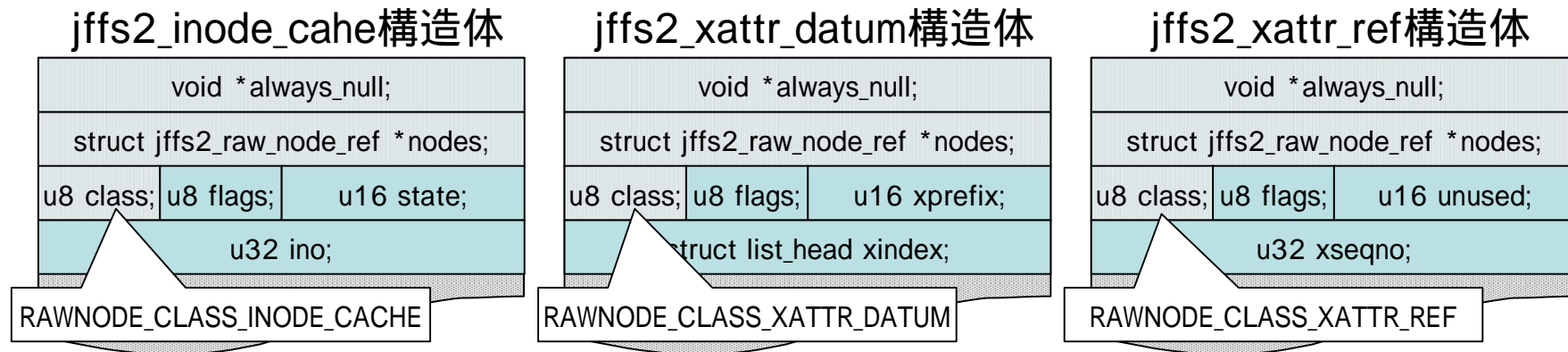
XATTR on JFFS2 のデータ構造 (4)



XATTR on JFFS2 のデータ構造 (5)

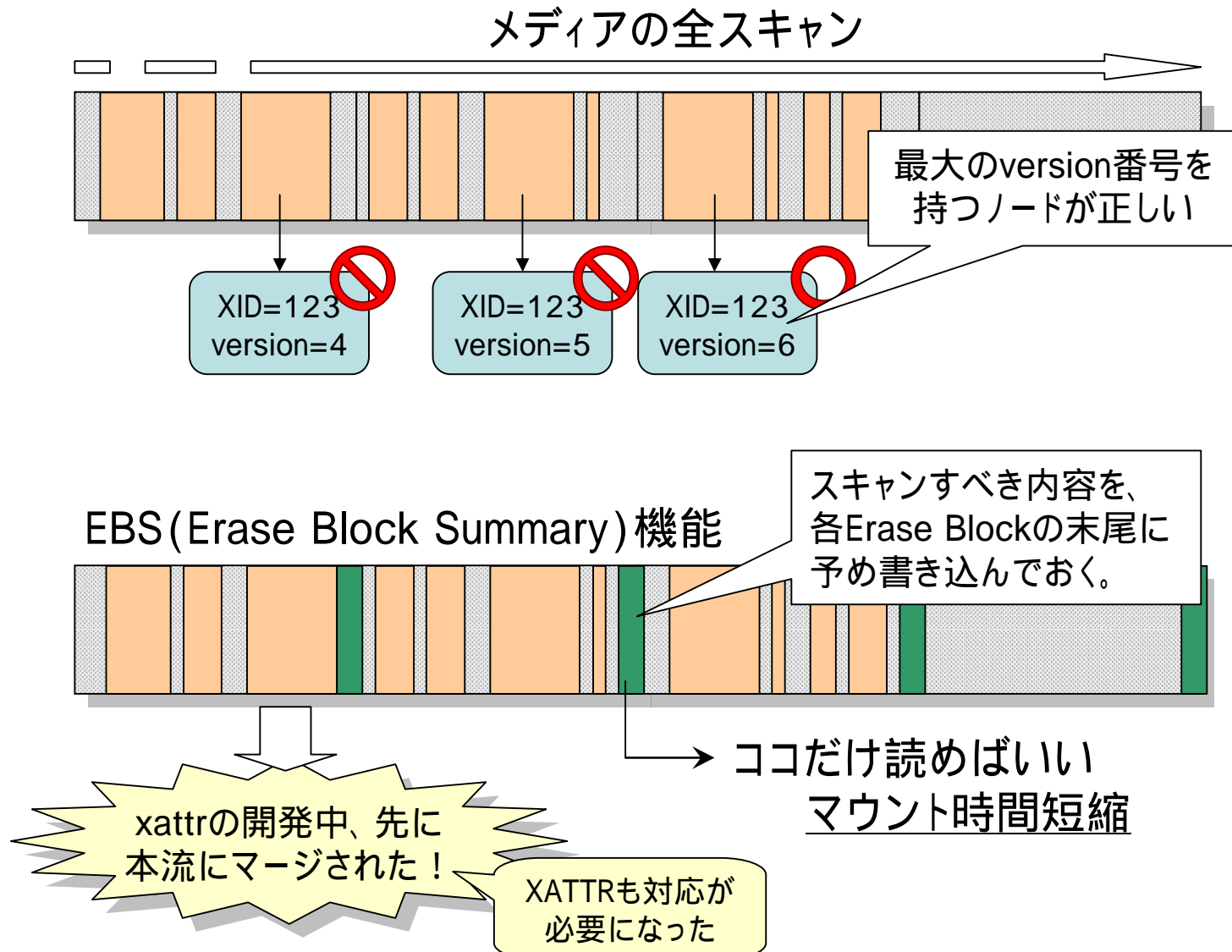


XATTR on JFFS2 のデータ構造 (6)

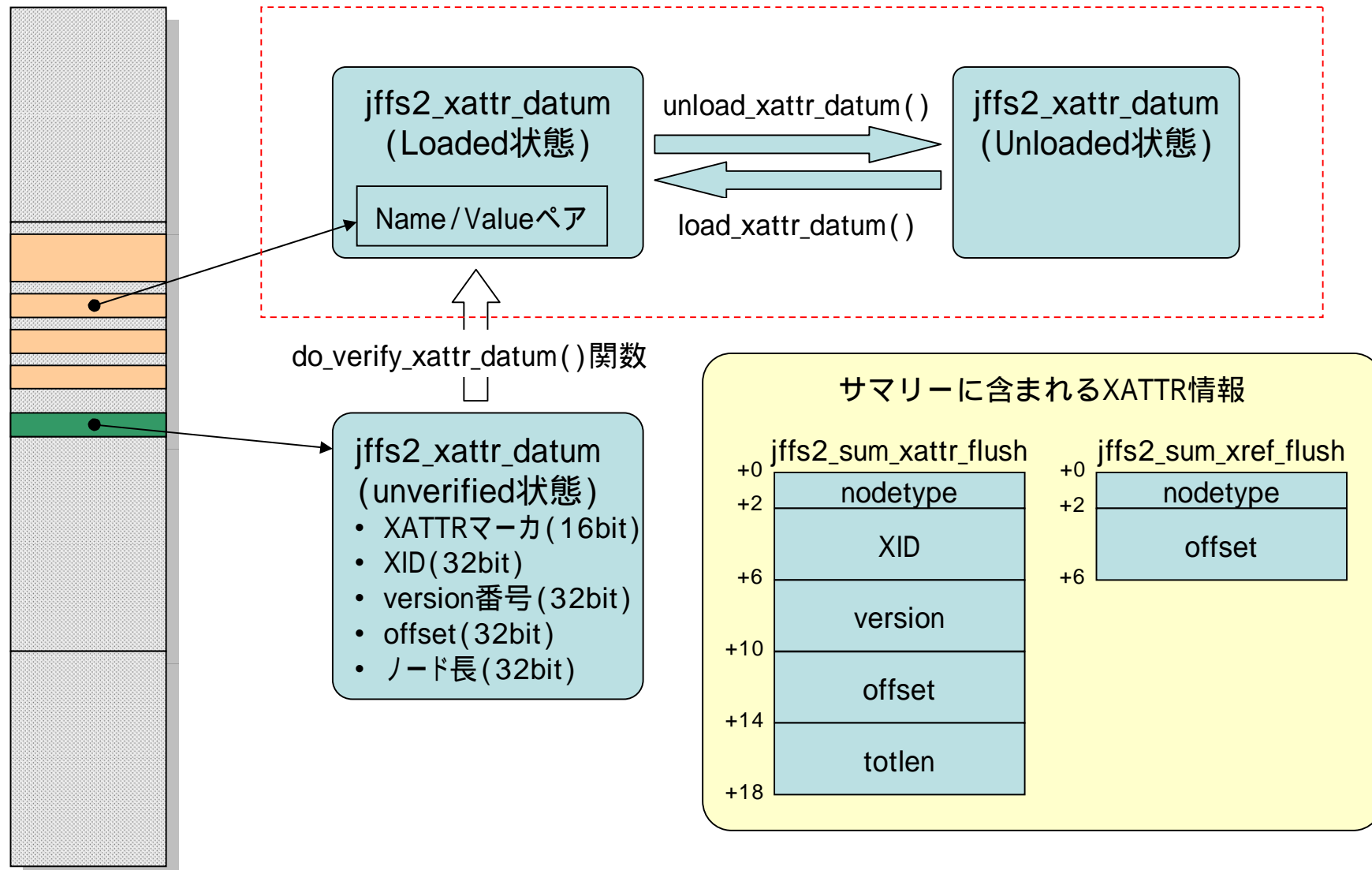


- jffs2_raw_node_refオブジェクトは非常に多く生成される
 - 構造体のサイズを増加させることは許されない。
- 構造体の頭の部分は共通のフォーマット
 - class変数を参照すれば、オブジェクトの型を特定できる。
 - 元々は32bit幅あったstate変数を、16bitに縮小して確保した領域
jffs2_inode_cache構造体のサイズは、jffs2_xattr_refへのポインタ分だけの増加で済んだ。(+4byte)
 - ガベージコレクタの処理を大部分共通化することを可能に。

マウント時の処理(1)



マウント時の処理(2)



目 次

- イン트로ダクション
- XATTR / JFFS2の構造
- コミュニティからのフィードバック
- XATTR / JFFS2の活用と今後

ノードを削除する時の処理(1)

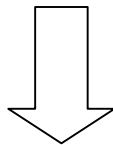
- 海外による当初の実装(~ revision.5)では
 - xattr_datum, xattr_refは常に一個しかノードを持たない
 - ノードを削除するときには、ノードのヘッダを0xffで上書き
 - これはNAND型フラッシュメモリでは使えないと指摘
- "delete marker"を使った実装(revision.6)に変更
 - 「このXATTRは削除されています」というマークを、最も大きなバージョン番号を持つノードに持たせる
 - 次回マウント時に、このXATTRはスキップされる
 - XATTR削除のアトミック性の問題
 - セットしたXATTRが"削除済み"になってしまう

ノードを削除する時の処理(2)

- XATTR削除のアトミック性問題
 - どのinodeからも参照されないXATTRを「削除済み」として扱うよう修正

XATTRを削除する処理

~~JFFS2_NODETYPE_XATTR
with 削除マーカの書込み~~



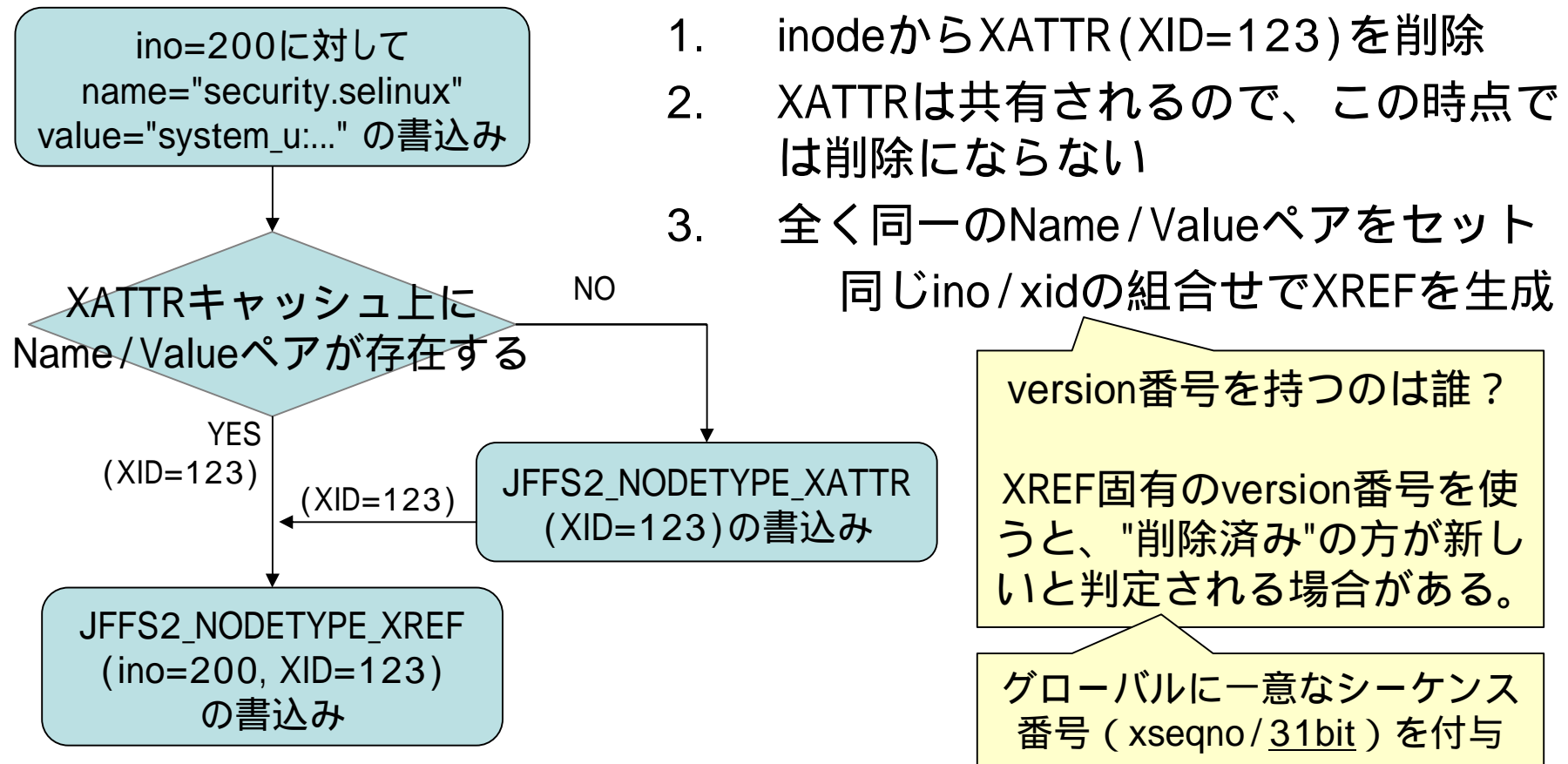
JFFS2_NODETYPE_XREF
with 削除マーカの書込み

- XREFを書き込む前に、
メディアが一杯になったら？
OSがクラッシュしたら？

JFFS2は元々、マウント時にメディア
上の全ノードをスキャンする
「XATTRを参照するXREFの個数」を
削除フラグとして利用すればよい

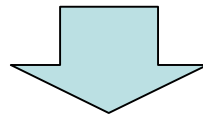
ノードを削除する時の処理 (3)

- 書き込んだXATTRが"削除済み"になってしまう



David Woodhouse氏のリファクタリング

- jffs2_raw_node_refのサイズ削減(16byte → 8byte)
 - 物理ノードを獲得するインターフェースが変更
- jffs2_inode_cacheのフォーマット変更
 - G Cの実装をinode / xattr / xrefで共通化



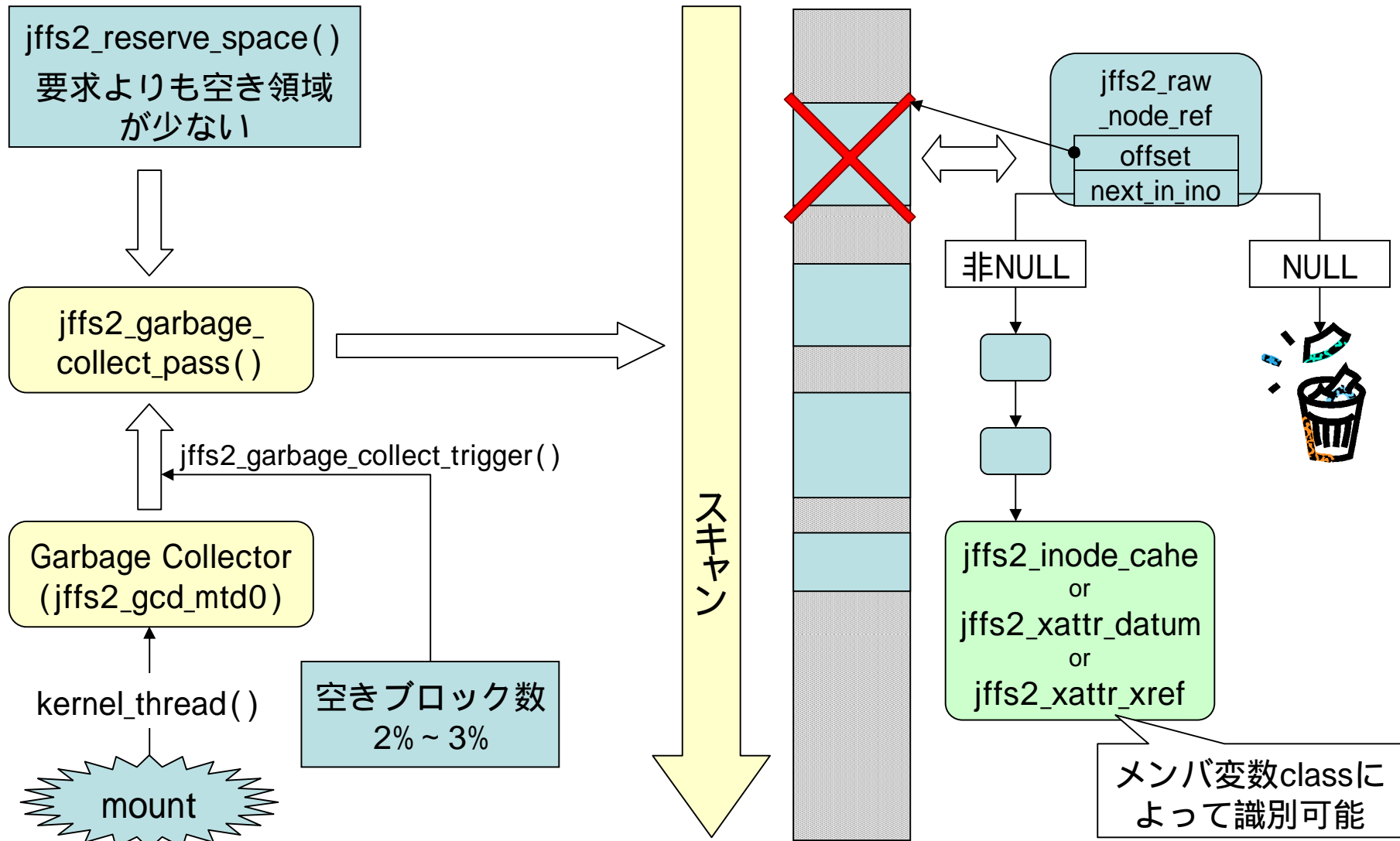
inodeの部分は修正したから、xattr, xrefの実装を新しいフォーマットに合わせてね。



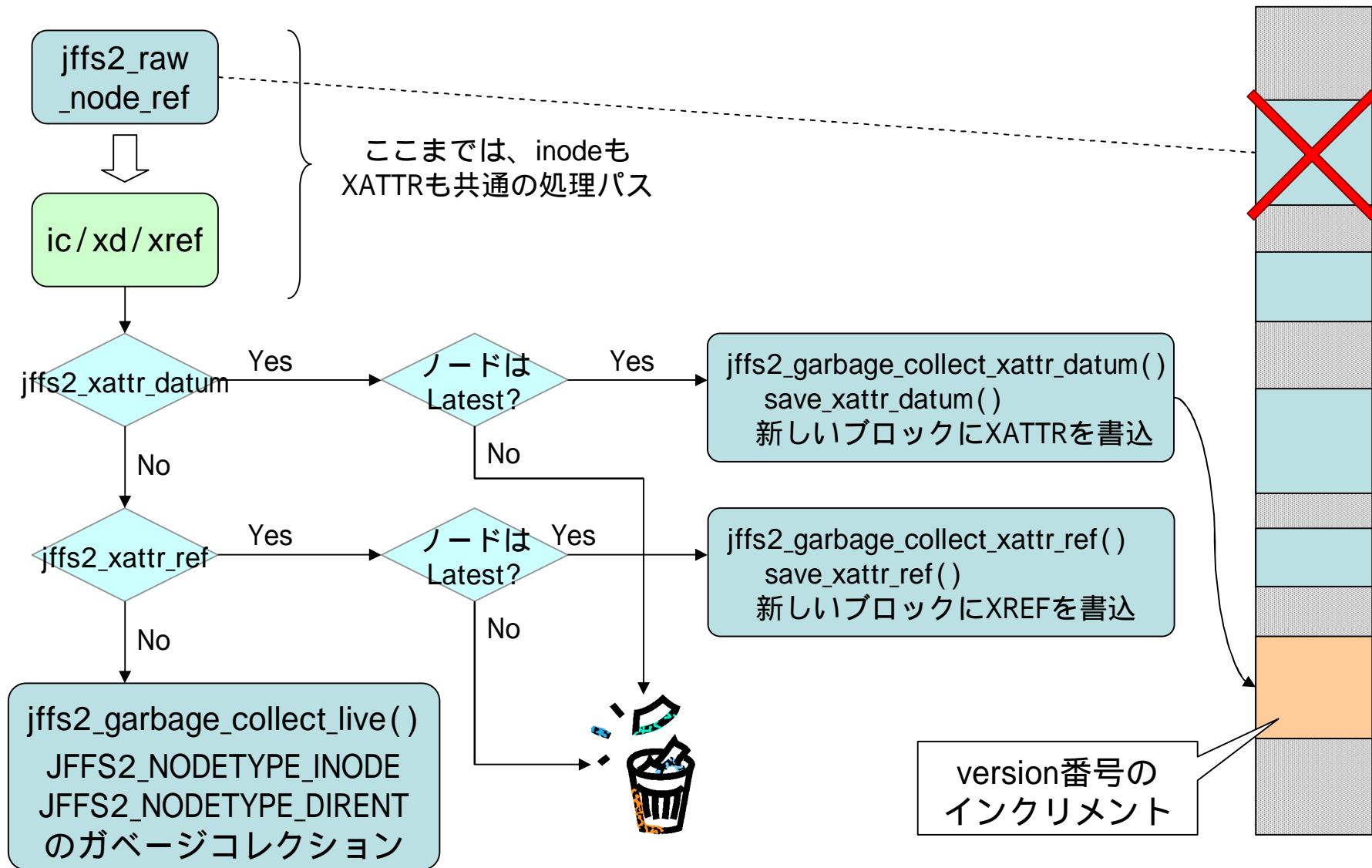
David
Woodhouse

ファイルシステム全体を俯瞰した
より見通しのよいコーディングに

ガベージコレクション(1)

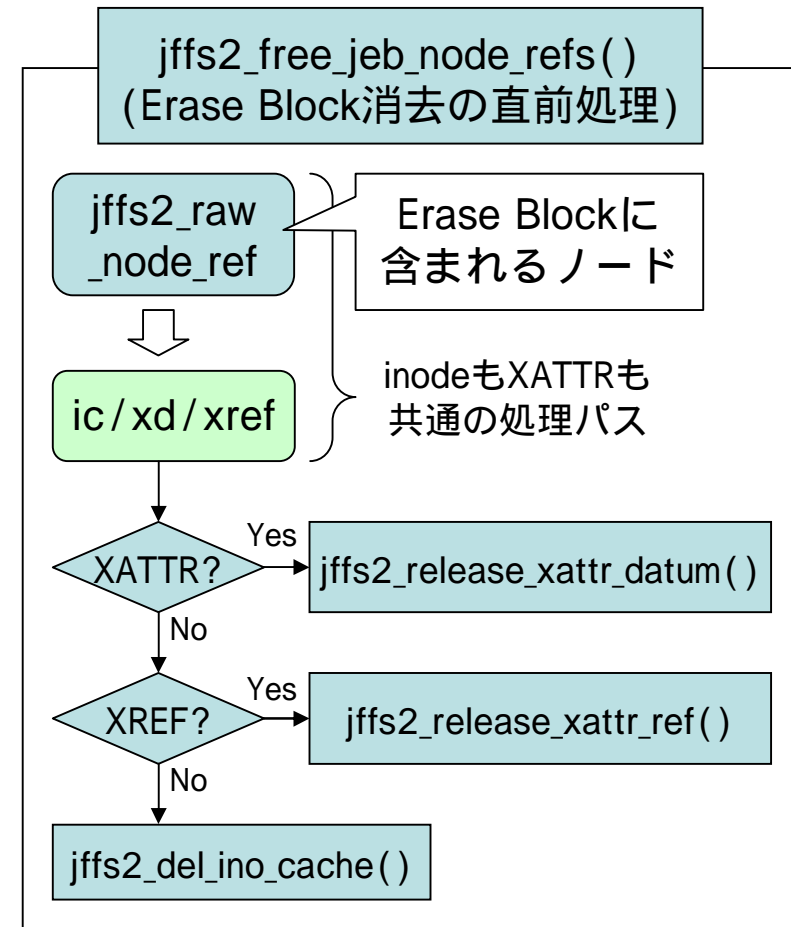
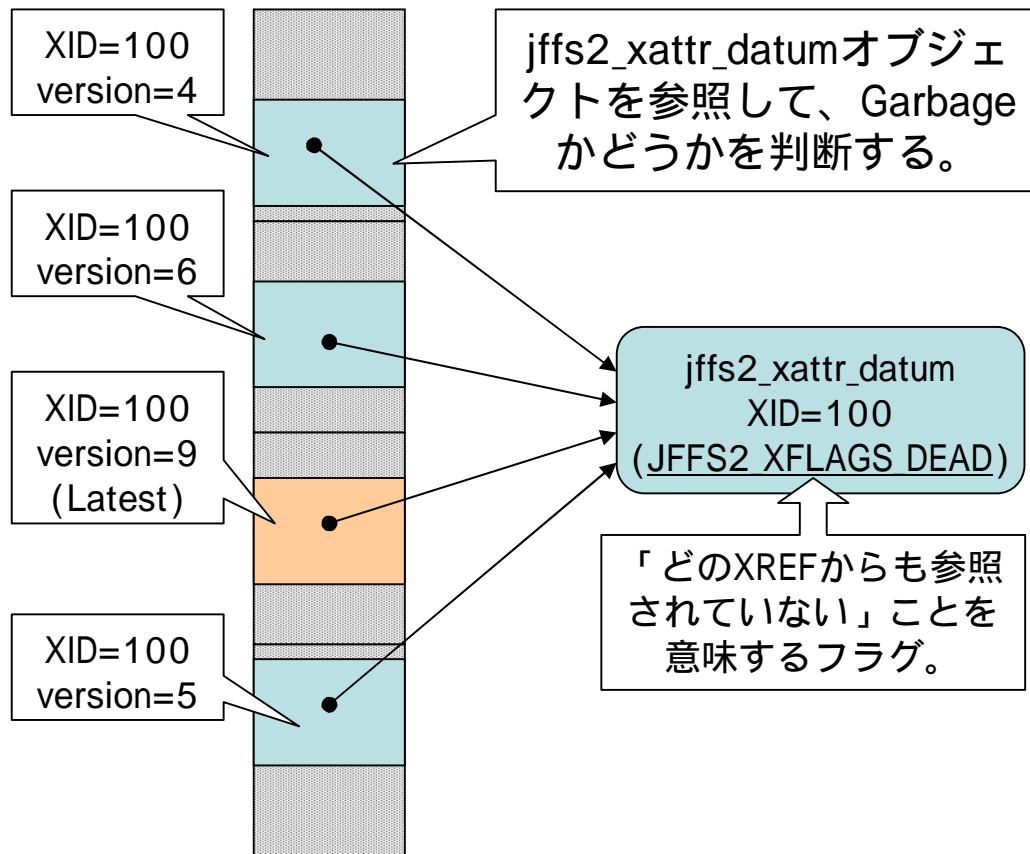


ガベージコレクション(2)



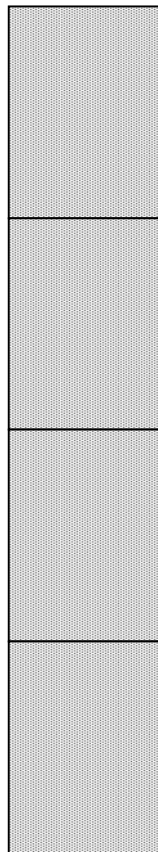
ガベージコレクション(3)

`jffs2_xattr_datum`
`jffs2_xattr_ref` } は、即座にfreeできない。



ガベージコレクション(3)

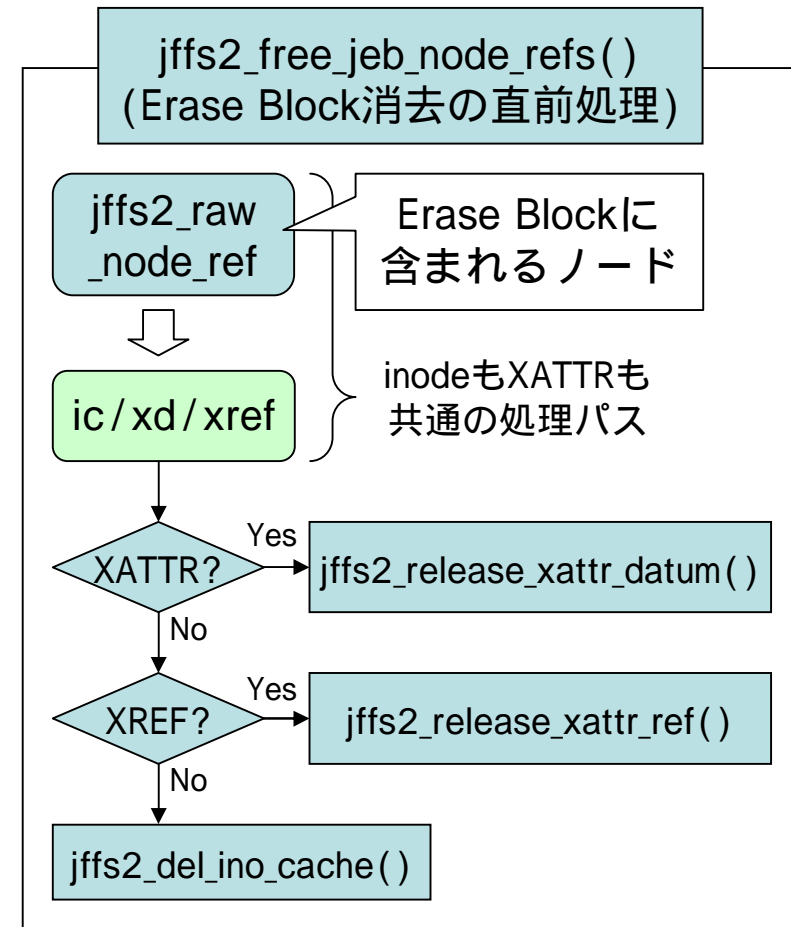
`jffs2_xattr_datum`
`jffs2_xattr_ref` } は、即座にfreeできない。



参照するノードが全て
消去されたので、freeす
ることが可能。

`jffs2_xattr_datum`
XID=100
(JFFS2 XFLAGS DEAD)

「どのXREFからも参照
されていない」ことを
意味するフラグ。



それは要らないんじゃないの？

- XATTRのName / Valueペアの長さ制限

```
#define XATTR_NAME_MAX    255
#define XATTR_SIZE_MAX    65536
```

- XATTR / JFFS2では、EraseBlockの大きさに依存
(NOR-Flash : 64KB - 128KB、 NAND Flash : 8KB - 16KB)
- 「複数ノードにまたがったXATTRを実装できない？」
- 「それって本当に必要なの！？」
 - SELinuxやPOSIX-ACLでは、高々100byte程度で十分
 - Ext2 / 3でも、XATTRの総サイズはBlock-Size (4Kbyte)
- 「Ext2 / 3でもその程度なら、まあ問題ないか。」

2.6.18カーネルでXATTR / JFFS2がマージ

[projects](#) / [linux/kernel/git/torvalds/linux-2.6.git](#) / [commit](#)

[summary](#) | [shortlog](#) | [log](#) | [commit](#) | [commitdiff](#) | [tree](#)

[JFFS2][XATTR] XATTR support on JFFS2 (version. 5)

author KaiGai Kohei <kaigai@ak.jp.nec.com>
Sat, 13 May 2006 06:09:47 +0000 (15:09 +0900)
committer KaiGai Kohei <kaigai@ak.jp.nec.com>
Sat, 13 May 2006 06:09:47 +0000 (15:09 +0900)
commit aa98d7cf59b5b0764d3502662053489585faf2fe
tree e98e83f3e69ebe3a1112394a19d440419e899749 [tree](#)
parent 4992a9e8886b0c5ebc3d27eb74d0344c873eeea [commit](#) | [commitdiff](#)

[JFFS2][XATTR] XATTR support on JFFS2 (version. 5)

This attached patches provide xattr support including POSIX-ACL and SELinux support on JFFS2 (version.5).

There are some significant differences from previous version posted at last December.

The biggest change is addition of EBS(Erase Block Summary) support. Currently, both kernel and usermode utility (sumtool) can recognize xattr nodes which have JFFS2_NODETYPE_XATTR/_XREF nodetype.

In addition, some bugs are fixed.

- A potential race condition was fixed.
- Unexpected fail when updating a xattr by same name/value pair was fixed.
- A bug when removing xattr name/value pair was fixed.

The fundamental structures (such as using two new nodetypes and exclusion mechanism by rwsem) are unchanged. But most of implementation were reviewed and updated if necessary.

Espacially, we had to change several internal implementations related to load_xattr_datum() to avoid a potential race condition.

[1/2] xattr_on_jffs2.kernel.version-5.patch
[2/2] xattr_on_jffs2.utils.version-5.patch

Signed-off-by: KaiGai Kohei <kaigai@ak.jp.nec.com>

Signed-off-by: David Woodhouse <dwmw2@infradead.org>

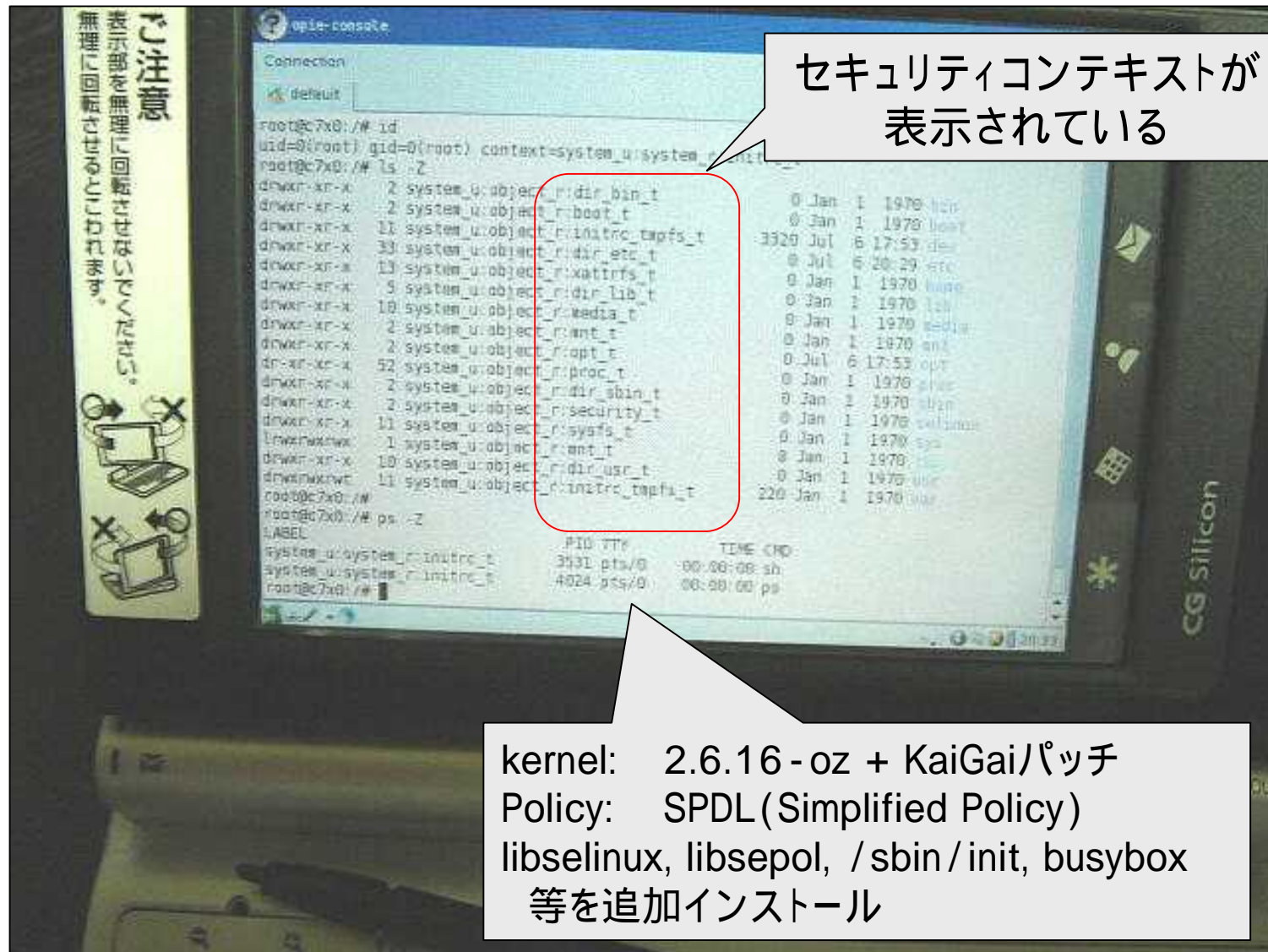
その他、2.6.18カーネルにはJFFS2 / XATTRの実装に合計20本のパッチがマージ

[JFFS2] [XATTR] Fix memory leak in POSIX-ACL support
[JFFS2] [XATTR] Fix xd->refcnt race condition
[JFFS2] [XATTR] coexistence between xattr and write ...
[JFFS2] [XATTR] Fix wrong copyright
[JFFS2] [XATTR] Re-define xd->refcnt as atomic_t
[JFFS2] [XATTR] Fix memory leak with jffs2_xattr_ref
[JFFS2] [XATTR] rid unnecessary writing of delete marker.
[JFFS2] [XATTR] Fix ACL bug when updating null xattr ...
[JFFS2] [XATTR] using 'delete marker' for xdatum/xref ...
[JFFS2] [XATTR] Fix obvious typo
[JFFS2] [XATTR] Handling the duplicate JFFS2_NODETYPE ...
[JFFS2] [XATTR] remove redundant pointer cast in acl.c
[JFFS2] [XATTR] remove '__KERNEL__' from acl.h
[JFFS2] [XATTR] remove senseless comment
[JFFS2] [XATTR] Unify each file header part with any ...
[JFFS2] [XATTR] '#include <linux/list.h>' was added ...
[JFFS2] [XATTR] Remove jffs2_garbage_collect_xattr(c, ic)
[JFFS2] [XATTR] Remove 'struct list_head ilist' from ...
[JFFS2] [XATTR] Add a description about c->xattr_sem
[JFFS2] [XATTR] remove typedef from posix_acl related ...
[JFFS2] [XATTR] XATTR support on JFFS2 (version. 5)

目 次

- イン트로ダクション
- XATTR / JFFS2の構造
- コミュニティからのフィードバック
- XATTR / JFFS2の活用と今後

OpenZaurusにSELinuxを入れたみた



OLPC (One Laptop Per Child) プロジェクト

- 発展途上国の子供たちにPCを！
 - Digital Divideの解消
 - Linuxの普及・啓発
- \$100PC
 - 100万台のオーダー × 数カ国
 - 記憶デバイスは512MBのFlash-ROM + JFFS2
 - セキュリティ強化にSELinuxの利用
 - 多数の同一構成マシンが出るので、0-day攻撃は超脅威
 - パッチ適用が難しい環境での利用もあり得る
 - SELinuxの利用には、JFFS2上でのXATTRのサポートが必須



今後の課題

- JFFS2向けPOSIX - ACLのメモリ使用量削減
 - Ext2 / 3のPOSIX - ACLを参考にして実装
 - JFFS2のXATTRはname / valueペアをキャッシュできる
- busybox向けSELinuxコマンドの開発
 - chcon, setsebool, setenforce 等々
- 組み込み分野に特化したポリシーの開発
 - Reference Policyはサーバ利用を前提
 - ポリシーサイズの削減は attribute の利用が鍵
 - サーバ用でないディストロ向けのセキュリティポリシー

Any Question?



付録 ～ XATTR/JFFS2 開発の経過 ～

- Revision.1 ('05/08/23)
 - XATTR/JFFS2のプロトタイプ。GCは未実装。
- Revision.2 ('05/09/07)
 - XATTRノードに対するGCのサポートを追加。
- Revision.3 ('05/09/28)
 - Revision.2でのデッドロックのバグを修正したバージョン
- Revision.4 ('05/12/28)
 - Posix-ACLのサポートと、2.6.14カーネルでのXATTR初期化に対応
- Revision.5 ('06/05/09)
 - EBS(Erase Block Summary)サポートを追加
 - gitツリーのベースとなる
- Revision.6 ('05/06/24)
 - "Delete Marker"によるノードの削除に変更